

Софийски Университет „Св. Климент Охридски“  
Стопански факултет

**Дисциплина**  
**“Управленски информационни системи”**

**КУРСОВА РАБОТА**

**НА ТЕМА**

**WEB SERVICES, SOA  
И ТЯХНОТО ПРИЛОЖЕНИЕ**

Преподавател: ас. А.Антонова

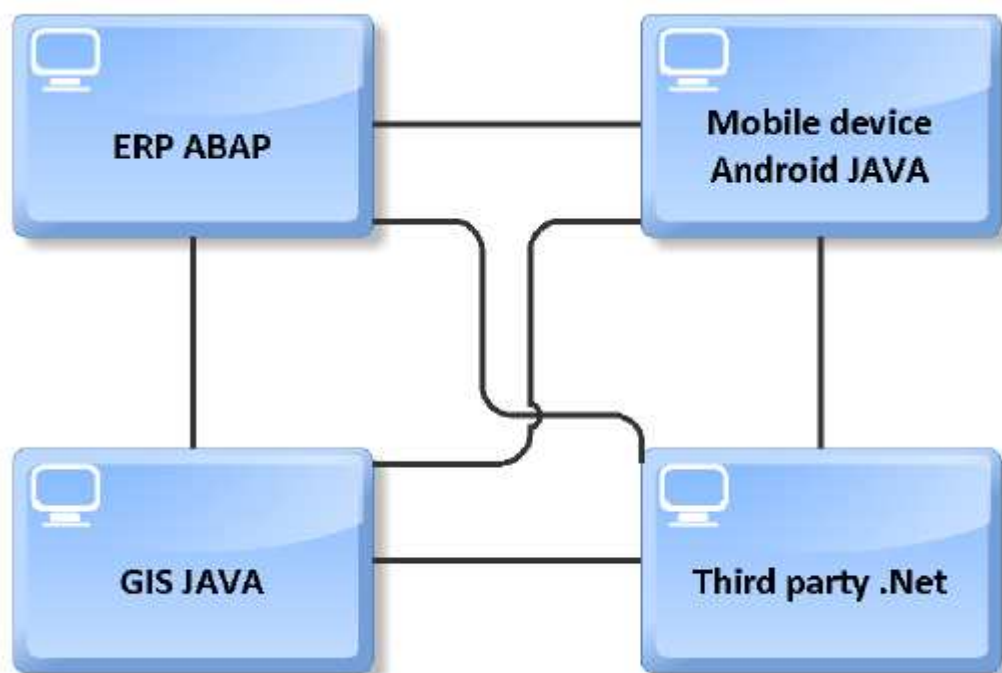
Дата: 27.12.2011  
Изготвил: Станислав Назиков  
Факултетен номер: 10783

**СЪДЪРЖАНИЕ**

|       |  |        |
|-------|--|--------|
| 1     | Въведение.....                                 | - 3 -  |
| 2     | Теоретична част.....                           | - 5 -  |
| 2.1   | Въведение в уеб услугите.....                  | - 5 -  |
| 2.2   | Основни компоненти на Уеб услугите.....        | - 7 -  |
| 2.2.1 | Подател на услугата (service provider).....    | - 9 -  |
| 2.2.2 | Косуматор на услугата (service consumer) ..... | - 9 -  |
| 2.2.3 | WSDL език .....                                | - 9 -  |
| 2.2.4 | SOAP.....                                      | - 10 - |
| 2.2.5 | UDDI.....                                      | - 11 - |
| 3     | Практическа част.....                          | - 12 - |
| 3.1   | Профил на компанията и бизнес казус.....       | - 12 - |
| 3.2   | Решението .....                                | - 14 - |
| 3.3   | Бизнес изглед (Business view) .....            | - 18 - |
| 3.4   | Процесен изглед (Process view).....            | - 19 - |
| 3.5   | Приложен изглед (Application view).....        | - 20 - |
| 3.6   | Технически изглед (Technical view) .....       | - 20 - |
| 4     | Заключение .....                               | - 21 - |
| 5     | Референции .....                               | - 22 - |

## 1 Въведение

През по-голямата част на 20-и век до наши дни развитието на информационните технологии е безусловен факт. Те са навсякъде в нашето ежедневие, подпомагащи ни в работата или в личния живот. Използваме информационни системи в комуникациите по мобилен телефон, използваме информационни системи, за да изпращаме и получаваме електронна поща, използваме информационни системи, за да получаваме информация, която ни е нужна. Те са такава неизменна част от живота ни, че годишно се създават множество софтуерни продукти и платформи обслужващи различни наши нужди, системи с различни функционалности. В повечето компании съществуват по няколко системи – за управление на склада, за счетоводство, за човешки ресурси и то на различни производители. При наличието на толкова много системи възниква въпросът за взаимодействието между тях, за начина по който самите информационни системи могат да работят една с друга, да си комуникират и да предоставят една обща информация на своите потребители и то по оптимален начин. Но всяка една информационна система има свои специфики – има различни операционна системи и софтуер, който може да функционира само на тези платформи. Това поставя въпроса за интеграцията на системите. На фигурата се илюстрира комплексността на връзката между системите, като разбира се връзките могат да бъдат много повече:



Много от доставчиците на информационни системи, в частност ERP системи, дори доставят собствени платформи за интеграция. Така например SAP AG предоставя SAP Exchange Infrastructure (XI), за да може да се извършва интеграцията между ERP системата и външни информационни системи. Те предоставят и собствени инструменти (Integration builder), с които да се моделират връзките между системата изпращач и получател на данните. По този начин те подsigуряват възможността за връзка с множество различни софтуерни системи, независимо дали GIS системи, други ERP системи, CRM и пр. Нараства и делът на така наречените системни интегратори или компании, които се занимават изключително с интегриране на различни видове софтуер. Има различни middleware решения подобни на това на SAP, чиято цел освен техническата интеграция е и да спестят финансови средства на компаниите. Интеграцията между различни приложения е предизвикателство доколкото се свързват два или повече напълно различни като функции и като технически характеристики софтуерни и хардуерни компоненти. Когато се имплементират такива решения, винаги има голям риск, именно заради тези разлики. Цел на тази курсова работа е да разгледа една технология позволяваща интеграция на системи на различни доставчици и точно комуникация между системи, които софтуерно нямат нищо общо една с друга. Тази технология се нарича Web Services и самата платформа Service Oriented Architecture (SOA). Независима е от софтуерните платформи и различните доставчици на приложения, тъй като се базира на всеобщия стандарт XML и на стандартни протоколи за обмен на данни. Концепцията за SOA не е нова. Компанията SUN я използват още през 1990 г. а в момента това е водещата технология в тази област. Лидери като ORACLE, IBM, SAP са насочили своите усилия в разработването на решения в тази област, за да позволят лесна и ефективна интеграция със собствения си софтуер. И за да бъде улеснено разбирането за web services ще започнем с един елементарен пример: да предположим, че съществува приложение за навигация с карти, което има списъци с улици и адреси, но се нуждае и от GPS координати срещу тях – но няма GPS функционалност. Да предположим, че съществува и приложение за GPS навигация, което има имплементирана функционалност при подаване на точен адрес, то връща като отговор точните GPS координати. Ако тази функционалност бъде реализирана с помощта на web service, то първото приложение може да се интегрира с второто, така че да получава необходимите му GPS координати срещу подаден от него адрес.

## 2 Теоретична част

### 2.1 Въведение в уеб услугите

SOA и web services са две различни неща, но web services са начин, по който да се схване концепцията на SOA. SOA е архитектурен стил за изграждане на софтуерни приложения, които използват мрежови услуги. Тези приложения се базират на услугите. Услугата от своя страна е имплементация на добре дефинирана бизнес функционалност, като тя може да бъде консумирана от различни клиенти с различни приложения, т.е. web services са независими. Според World Wide Web Consortium (W3C) SOA е сбор от компоненти, които могат да бъдат извиквани и чиито интерфейси могат да бъдат публикувани, което определя най-важната част от SOA, а именно разделянето на интерфейса на услугата от нейната имплементация. Или ако трябва да обобщим всичко това с по-прости думи: web service е „изнасяне” на една функционалност на системата навън, така че да може да бъде ползвана от други системи. Общоприета дефиниция за web services все още липсва. Общото между всички представи за Уеб услугите е, че става дума за споделяне на ресурси чрез Уеб - с помощта на XML по HTTP. Повечето технологии и концепции за Уеб услуги са все още са в процес на стандартизиране и продължават споровете около тяхната същност. В това число, все още няма единна концепция за архитектурата на стака на Уеб услугите. Понякога дори се поставя знак на равенство между Web services и Application services. И в двата случая става дума за споделяне на ресурси по мрежата - в единия случай на данни или изчислителна мощ, в другия - на различни приложения. Application services означава предоставяне на различни програми "до поискване". Общата тенденция в развитието на високите технологии е, че те вървят към тотална глобализация и в не много далечно бъдеще сегашните РС и други компютърни системи ще бъдат заменени от мрежови устройства. И тогава няма да имаме нужда от свое локално копие на MS Word, например. Ще си имаме свой ASP (Application Services Provider), който ще доставя тази услуга, а ние ще я ползваме. Издигайки се над противоречията, можем да очертаям общата представа и смисъл на Уеб услугите като опит за създаване на по-висше ниво Уеб чрез интегриране на мощта и данните, съхранявани на отделни машини. За да се получи такава интеграция, обаче, е необходимо да се намери общ език между тях, без значение на какъв програмен език са написани отделните приложения, в каква операционна среда функционират, кой им е

производител и какво е предназначението им. Ключ към създаването на такъв общ език е XML, който стои в основата на концепцията за Уеб услугите - той дава възможност на компютрите да комуникират и да се разбират без намесата на човека, структурирайки информацията по строго определени начини. Целта на Уеб услугите е да се преодолеят различията между езиците, както и технологиите и продуктите на отделните производители, така че всички единици в мрежата да могат да комуникират пълноценно. Могат да се пишат приложения за Уеб услуги на всеки един програмен език, който има възможности за обработка на XML. В това число влизат VC++, C#, VB, Java, JavaScript, PHP, ASP и така нататък. Всичко зависи от сложността на услугата, която трябва да се разработи, тъй като различните езици имат различен потенциал. Важното е да бъде осигурен фронтенд на приложението, който говори и разбира SOAP и WSDL - общите езици на Уеб услугите. Разбира се, необходими са ясно дефинирани стандарти и спазването на строги правила при предаването на информацията, за да се осигури пълната интеграция и разбиране между различните приложения. Основните стандарти на които се опират Уеб услугите са SOAP и WSDL. Но преди да преминем към стандартите ето и някои практически примера за използването на web services

- ✓ **Интеграция на back-end системи:** International Truck използва web services за интеграция на множество информационни системи, с които работи. Един от най-големите проблеми на компаниите е наличието на множество информационни системи с различни функционалности, но ползващи някои общи данни. Така например данни за клиентите трябва да има и в CRM системата и в счетоводния модул на ERP системата. От разгледани 17 случая на фирми използващи web services, 6 са свързани именно с връзки между системите.
- ✓ **По-добри връзки с партньорите :** MedicAlert използва технологията за улеснена връзка между болници, лекари и фармацевти – партньори в здравната сфера. Тук от основно значение е използването на web services главно в B2B отношенията, когато партньорите се нуждаят от информация доставяна от фирмените системи, например клиенти, които искат да разгледат каталози с продукти или да създадат поръчки директно в системите на доставчика. От разгледани 17 случая на фирми използващи web services, 5 са свързани с подобряване връзките с доставчиците и 3 с клиентите – общо 8.
- ✓ **Оферирането на нови продукти:** Experian използва web services, за да позволи на клиентите си достъп до нови финансови услуги. Уеб услугите позволяват

лесна връзка през портал директно към корпоративните системи, извличащи информация оттам.

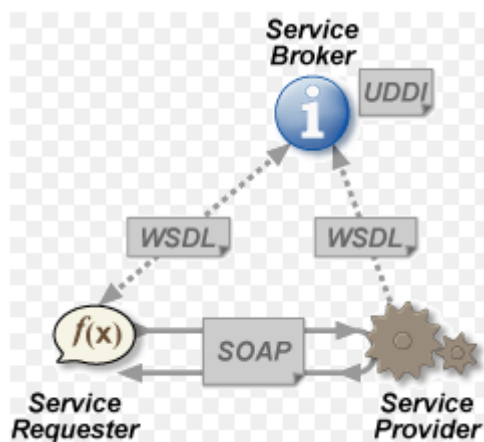
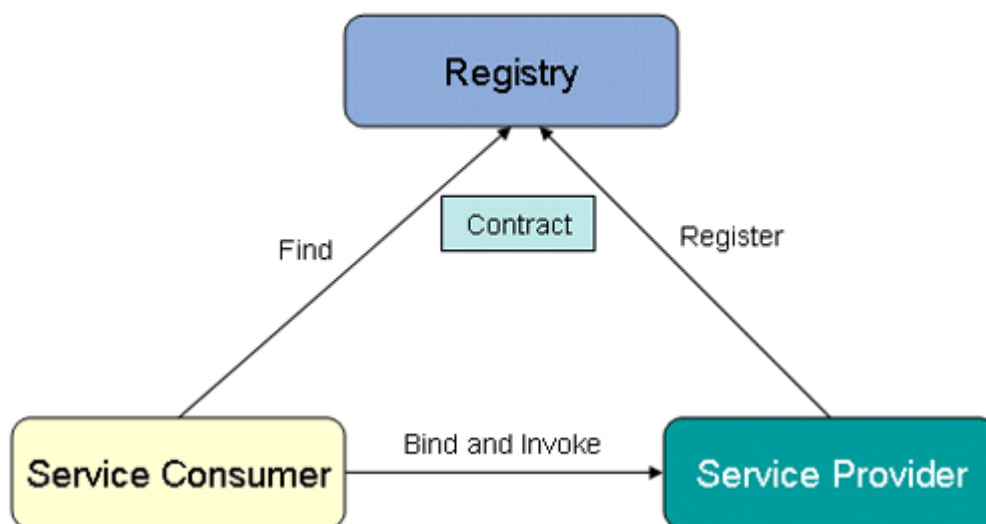
- ✓ **Обединяване на функционалности в едно:** Washington Group разработва единна front-end система, която комуникира с back-end системи чрез web services и използва функционалности от тях. Този случай е близък до първия, където става на въпрос за интеграция между корпоративните системи, но тук по-скоро интеграцията е с единна платформа за достъп, която обединява функции от останалите. Възможно е един бизнес процес да преминава през няколко информационни системи – тогава той се описва с BPEL език (Business process execution language), което позволява свързване с web services с тези системи
- ✓ **По-добро обслужване от IT отдела:** Siemens AG, използва технологията за връзка между отделите си и получаване на по-добро обслужване конкретно от своя IT отдел
- ✓ **Излизане на глобалния пазар:** Monster използва web services за да достигне до 24 нови страни в своя бизнес. Изполването на глобални стандарти при веб услугите е фактор за това да е без значени колко отдалечени са информационните системи физически една от друга. Чрез web service могат да се достъпят корпоративните информационни ресурси от всяка точка на планетата.

### ***2.2 Основни компоненти на Веб услугите***

Веб услугите комуникират с помощта на XML съобщения, написани според изискванията на SOAP спецификациите. SOAP съобщението представлява добре формиран и валиден XML документ. За валидация се използва XML Schema. SOAP е съкращение от Simple Object Access Protocol - протокол за отдалечен достъп до обекти. Най-често използвания метод за трансфер на SOAP съобщенията (които сами по себе си представляват, разбира се, чист текст) е по HTTP. SOAP е наследник на XML-RPC, тъй като представлява начин за извикване на процедури върху отдалечени системи. Само по себе си съществуването на една Веб услуга не е достатъчна предпоставка за нейното използване. Възниква проблемът как другите да разберат за съществуването на услугата, какво може да прави тя и ако да комуникират с нея. Очевидно е необходимо създаването на технология за индексване на услугите, която да даде възможност да търсим и откриваме необходимите ни. Тази технология е UDDI (Universal Description, Discovery and Integration). UDDI регистрите съдържат в себе си информация за Веб

услугите, компаниите, които ги поддържат, описание на възможностите и изискванията на отделните услуги. Те са нещо като телефонни указатели или "Жълти страници" за Уеб услуги. Но тук явно се сблъскваме и с нуждата от стандарт, с чиято помощ да се опише услугата. Това е WSDL (Web Services Description Language). WSDL документите описват в подробности технологичните особености на услугата - как да се свържем с нея, какви команди можем да ѝ подадем, какви параметри очаква тя от нас и какви ще ни върне, с какви типове информация работи и така нататък.

На следващите фигури са илюстрирани основните идеи и връзки при web services:





### 2.2.1 Подател на услугата (service provider)

Това е приложението, което притежава функционалност, от която се интересуват други приложения. За стандартите на SOA няма значение на какъв език е написано приложението и на каква платформа се изпълнява. Тази функционалност се описва на WSDL език и така може да бъде консумирана от други приложения.

### 2.2.2 Косуматор на услугата (service consumer)

Това е приложението, което консумира услугата предоставена от подателят. За него също няма значение на какъв програмен език е написано. То се интересува единствено от WSDL файла, за да получи „отговор” от подателя.

### 2.2.3 WSDL език

WSDL - език, използван за създаване на описание на услугата, която подателят предоставя, за това как се извиква и използва функционалността, която се крие зад web service. Описва местоположението и процедурите, които може да изпълнява, съобщенията, които приема и изпраща, а също и информация от по-високо ниво като данни за фирмата, хостваща и поддържаща услугата, ключови думи и т.н. Или по-подробно описани основни елементи във WSDL езика са : **types** (какви типове данни ще се прехвърлят), **message** (какво съобщение ще се прехвърля), **porttype** (какви функции ще се поддържат), **binding** (как ще се прехвърля съобщенията), **service** (къде се намира услугата).

Ето един пример как изглежда WSDL описанието:

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="HelloService"
  targetNamespace="http://www.ecerami.com/wsdl/HelloService.wsdl"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://www.ecerami.com/wsdl/HelloService.wsdl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <message name="SayHelloRequest">
    <part name="firstName" type="xsd:string"/>
  </message>
  <message name="SayHelloResponse">
    <part name="greeting" type="xsd:string"/>
  </message>
```

```

<portType name="Hello_PortType">
  <operation name="sayHello">
    <input message="tns:SayHelloRequest"/>
    <output message="tns:SayHelloResponse"/>
  </operation>
</portType>

<binding name="Hello_Binding" type="tns:Hello_PortType">
  <soap:binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="sayHello">
    <soap:operation soapAction="sayHello"/>
    <input>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:examples:helloservice"
        use="encoded"/>
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:examples:helloservice"
        use="encoded"/>
    </output>
  </operation>
</binding>

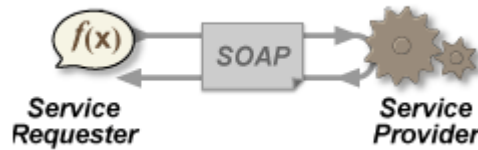
<service name="Hello_Service">
  <documentation>WSDL File for HelloService</documentation>
  <port binding="tns:Hello_Binding" name="Hello_Port">
    <soap:address
      location="http://localhost:8080/soap/servlet/rpcrouter"/>
    </port>
  </service>
</definitions>

```

Чрез WSDL файла се предоставя услугата и чрез него тя се консумира.

## 2.2.4 SOAP

SOAP - протокол, използван в комуникацията между Уеб услугите. Базиран на XML за съобщението и HTTP за трансфера му, той позволява на приложения от различни производители, написани на различни езици, да предават успешно данни и да извикват процедури в отдалечените системи. Като пример как се използва SOAP, може да се посочи, че съобщението се изпраща на web site, например за недвижими имоти, с параметър за търсене и тогава web site връща документ с резултата в XML формат (цени, местоположение и пр.). Всяко SOAP съобщение се състои от заглавна част (header) и тяло (body) и съдържа **Подател, Получател и Път на съобщението.**

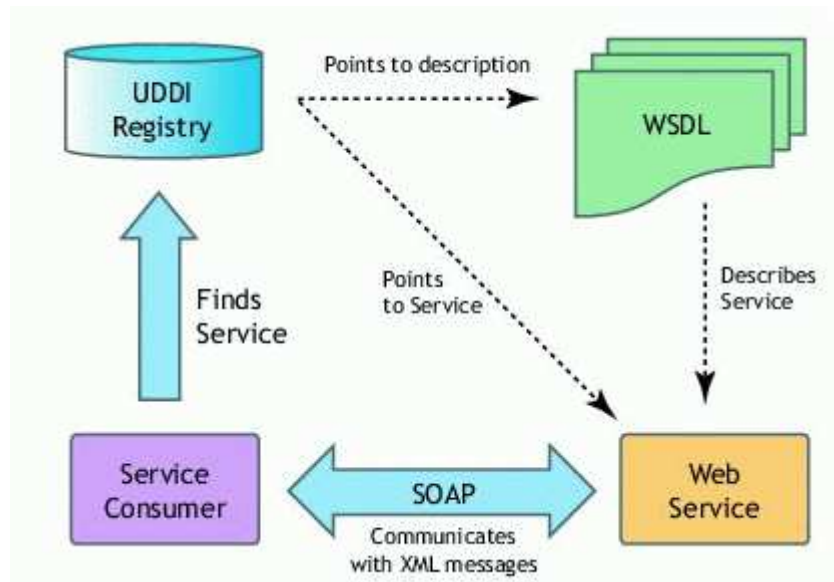


### 2.2.5 UDDI

UDDI - технология, използвана за съставяне на индекси и директории от Уеб услуги, които могат да бъдат претърсвани по определени критерии. UDDI регистрите могат да са публични или затворени (поддържани само в рамките на една компания или един проект).

След като е създаден един web service, то може да бъде публикуван в такъв UDDI регистър или контейнер. Или по точно казано публикува се неговия интерфейс – WSDL файла. Когато се намира в UDDI контейнера, WSDL файла може да бъде консумиран в зависимост от това ако UDDI е публичен, то от широката публика, ако е частен, то само от определени потребители (консуматори). Компании като SAP, Microsoft, IBM имат собствени UDDI регистри. Съществуват и доставчици на UDDI услуга, т.е. ако имате разработен web service, но нямате регистър може да ползвате такъв на друга компания.

На илюстрацията се вижда ролята на UDDI регистъра във вече познатата картинка:



### 3 Практическа част

#### 3.1 Профил на компанията и бизнес казус



Altova® е създател на продукти като XML SPY и други водещи XML, database и UML инструменти. Основана е през 1992 г. със седалище в Бевърли, Масачузетс и Виена, Австрия и е ключов играч на пазара на XML решения с над 4 млн. клиенти по целия свят. Активен член на W3C (World Wide Web Consortium) и доставя стандартизирани, независими от платоформите решения, лесни за употреба от потребителите.

Бизнес касусът, който се разглежда е на тази компания е че тя притежава доста широка продуктова гама като всеки продукт е в различен етап от цикъла на разработка. Цел на компанията е да намери начин да уведомява всички свои отдели, съответно служители за последните версии на своите продукти.

Управленският екип на Altova е създал база данни с информация за продуктите, така че да може да бъде използван от всички отдели (Продажби, Счетоводство, Поддръжка и др.) – така нареченият Каталог на продуктите. Чрез достъп до този каталог се получава бързо и достатъчна по обем информация за който и да е от софтуерните продукти. Екипът е решил, че най-удачния начин това да бъде изпълнено е като използва технологията на web service, за да дистрибутира информацията между отделите.

Разработката на web service е възложена на инженер на приложения, който работи в Altova от страната на крайните потребители. Неговият анализ на ситуацията идентифицира шест изисквания или задачи, които е нужно да бъдат изпълнени.

**Q: Как ще се нарича уеб услугата?**

Името и ще бъде : altovaCatalog

**Q: Къде ще се намира тя?**

Ще се намира в директория на вътрешен сървър: /services/altovaCatalog.

**Q: Как комуникират клиентите с Web service?**

Приложенията на клиентите ще изпращат и получават SOAP / RPC-кодирани съобщения. Първоначално ще се започне работа (ще бъде извиквана услугата) през уеб браузър, а на по-късен етап могат да бъдат добавени и други клиентски приложения освен уеб браузъра.

**Q: Какви операции ще извършва Web service?**

Ще има три основни операции:

- Да се достави каталог с категориите продукти при извикване
- Да се постави лист на продукти от която и да е категория
- Да се достави детайлизирана информация за всеки от продуктите

**Q: Какви параметри ще се подават на входа и какви резултати се очакват на изхода?**

| Операция \ I/O параметър                       | Параметър на входа | Резултат                                      |
|--|--------------------|---|
| Извикване на листа с категориите               | /                  | Лист на категориите дефинирани в базата данни |
| Извикване на лист от продукти в една категория | Категория продукти | Таблица с продуктите от тази категория        |

|  |                 |  |
|--|-----------------|--|
| Извикване на детайлизирана информация за даден продукт | Кода на продукт | Детайлизирана информация от базата данни |
|--|-----------------|--|

**Q: Какви data типове или data структури са нужни за всеки входен или изходен параметър?**

Data типовете ще се базират на структурата на вече съществуващите източници на данни.

### **3.2 Решението**

Ще бъде създаден WSDL файл, който да опише уеб услугата, като в него ще се опишат и структурите от данни, които ще бъдат трансферирани при операциите. Инженерът използва Altova XMLSpy за връзка с базата данни, където се намира информацията и автоматично генерира XML схема, която той след това оптимизира за уеб услугата.

Компонентите на схемата се дефинират като типове и параметрите, които ще се подават на SOAP съобщението се дефинират като глобални компоненти в йерархията на схемата.

Когато оптимизираната схема е завършена се създава WSDL файла отново чрез съществуващия инструментариум. След това се дефинират услугите, операциите и връзките с техните входни и изходни параметри. WSDL файлът се създава автоматично, докато инженерът работи в графичен режим на XMLSpy приложението. Този файл съдържа 175 реда (повече от 7000 байта), които по всяко време могат да бъдат превключени на текстов режим на обработка. Работата е изцяло чрез drag-drop функции и други визуални инструменти. Синтаксисът на WSDL файла се генерира автоматично

След това се използва SOAP менюто на софтуера за генериране на SOAP извикване от WSDL файла за всяка от трите операции. SOAP изикванията се запазват в XML файлове, за да бъдат използвани по-късно при тестване.

След като се даде отговор на всички шест въпроса WSDL файла се завършва. Чрез друг инструмент Altova MapForce се създава уеб услугата на базата на WSDL файла. Чрез него се създават и връзките с таблици и полета на базата данни към входните и

изходните параметри на уеб услугата. Връзките могат да бъдат тествани още по време на самата работа. Чрез SOAP съобщение се тестват запитвания към базата данни и се отчитат отговорите. След като бъдат навързани и трите операции инженерът генерира допълнителния source code на Java или C# и поставя скриптовете на уеб сървъра.

Уеб услугата се планира да се изпълнява на Apache Web server, така че се генерира Java source code, след което се компилира и поставя в действие.

След като уеб услугата е активна, инженерът я тества преди създаване на HTTP клиента. Това се извършва чрез XMLSpy SOAP menu и така може да се каже че уеб услугата е отделена от клиентското приложение.

За последната стъпка се разработва уеб страница чрез ASP scripting language, което всъщност е клиентския интерфейс. Разработва се XSLT stylesheet, за да се трансформира XML отговора от уеб услугата в HTML формат, така че да бъде изведен на уеб браузъра.

Завършената уеб услуга работи по следния начин:

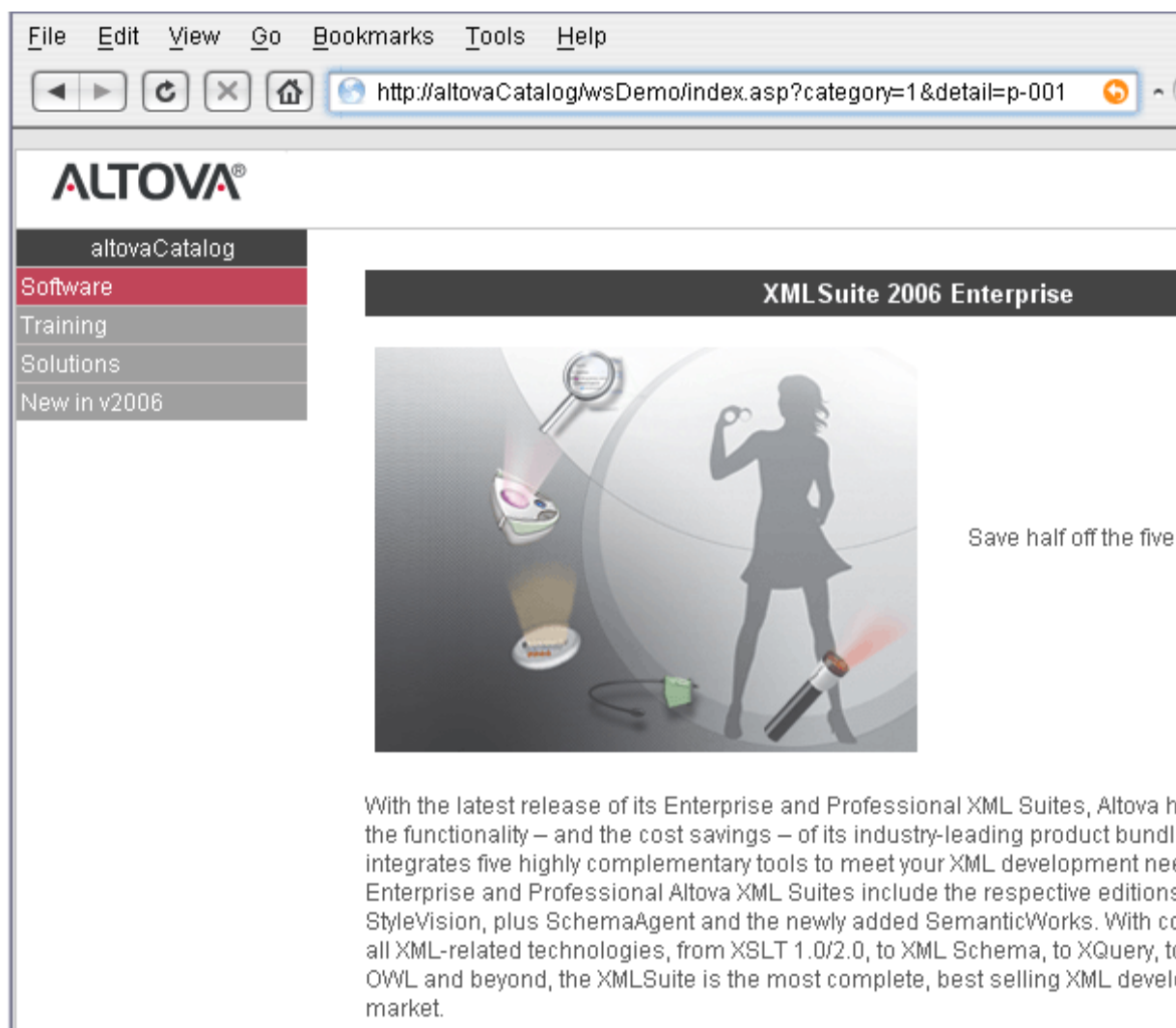
- Потребител отваря URL и израча запитване до уеб услугата за лист с категориите, които се появяват в горния ляв ъгъл на браузъра
- Кликване върху някоя от категориите води до таблица с продуктите съобразно селекцията
- Кликване върху някой от продуктите изпраща запитване за детайлизирана информация за всеки продукт, която се появява на мястото на таблицата.

На илюстрациите се вижда визуализацията през уеб браузър:

| sku   | name                          | edition      |
|-------|-------------------------------|--------------|
| p-001 | <a href="#">XMLSuite</a>      | Enterprise   |
| p-002 | <a href="#">XMLSuite</a>      | Professional |
| p-003 | <a href="#">XMLSpy</a>        | Enterprise   |
| p-004 | <a href="#">XMLSpy</a>        | Professional |
| p-005 | <a href="#">XMLSpy</a>        | Home         |
| p-006 | <a href="#">MapForce</a>      | Enterprise   |
| p-007 | <a href="#">MapForce</a>      | Professional |
| p-008 | <a href="#">MapForce</a>      | Standard     |
| p-009 | <a href="#">StyleVision</a>   | Enterprise   |
| p-010 | <a href="#">StyleVision</a>   | Professional |
| p-011 | <a href="#">SemanticWorks</a> |              |
| p-012 | <a href="#">SchemaAgent</a>   |              |
| p-013 | <a href="#">UModel</a>        |              |
| p-014 | <a href="#">DiffDog</a>       | Professional |
| p-015 | <a href="#">DiffDog</a>       | Standard     |
| p-016 | <a href="#">Authentic</a>     | Desktop      |
| p-017 | <a href="#">Authentic</a>     | Browser      |

Done

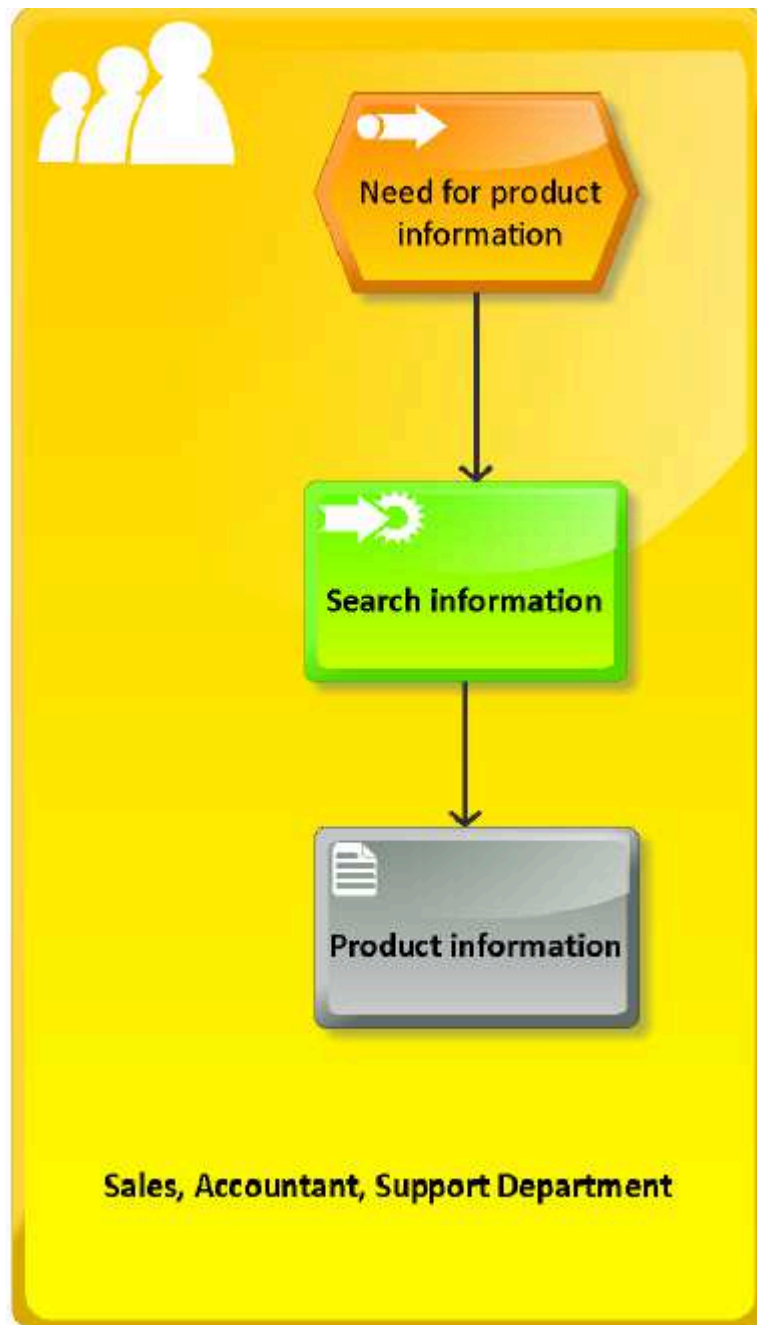




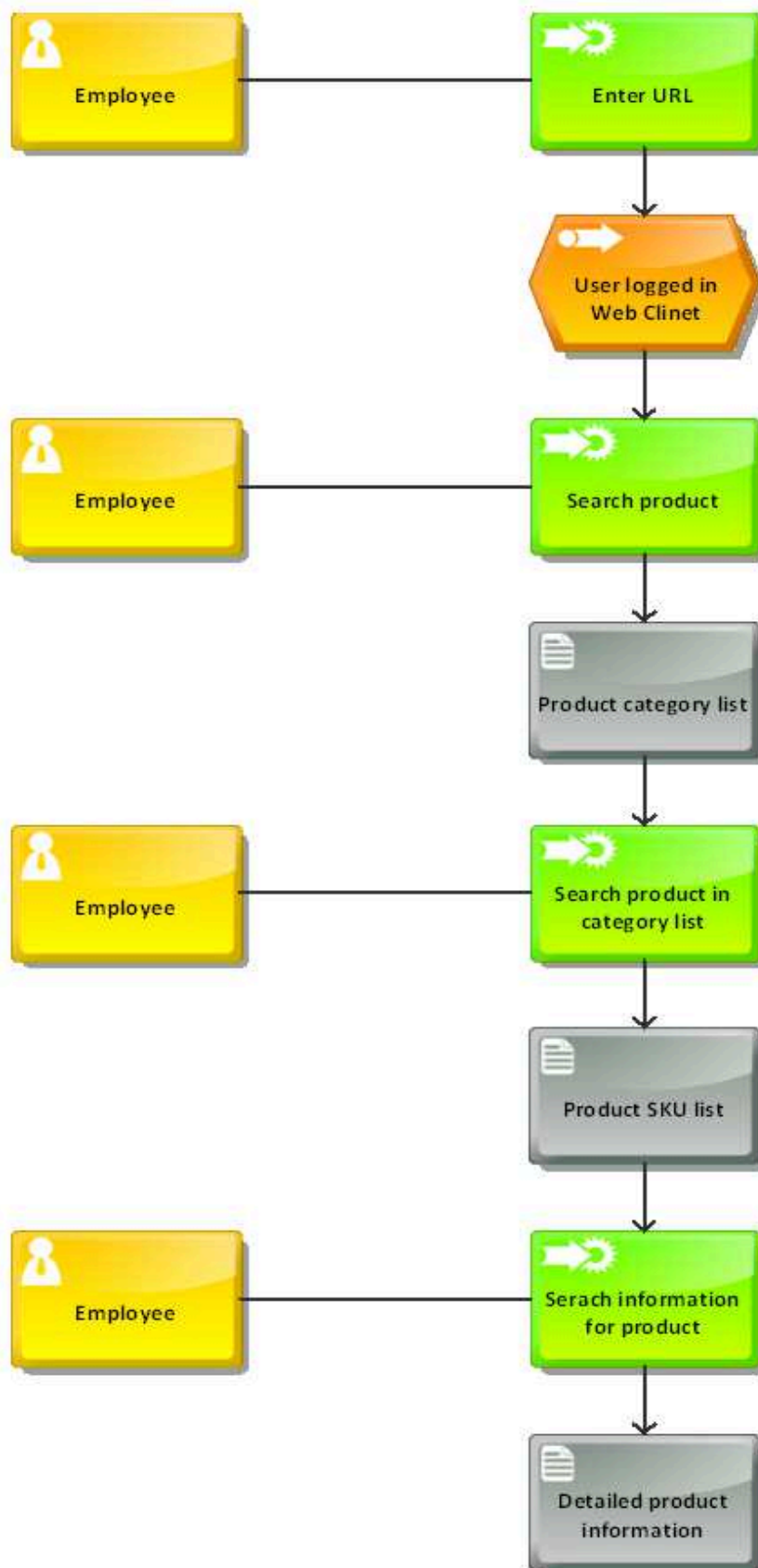
Ключово предимство при имплементацията на уеб услугата е, че когато мениджър добави нов продукт или версия в базата данни не се променя структурата на данните и уеб услугата, всеки път се изпълнява и извиква лист с категориите и не е нужна преработка на web service, като информацията е видима за всички отдели в компанията.

За да пусне в действие уеб услугата, продуктовият мениджър изпраща линк към каталога на всички отдели.

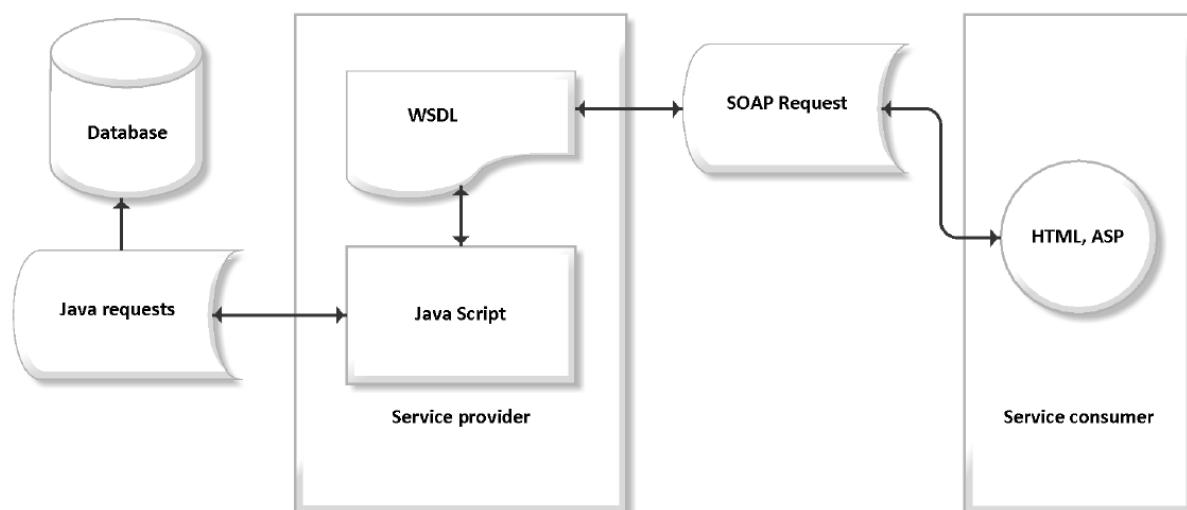
### 3.3 Бизнес изглед (*Business view*)



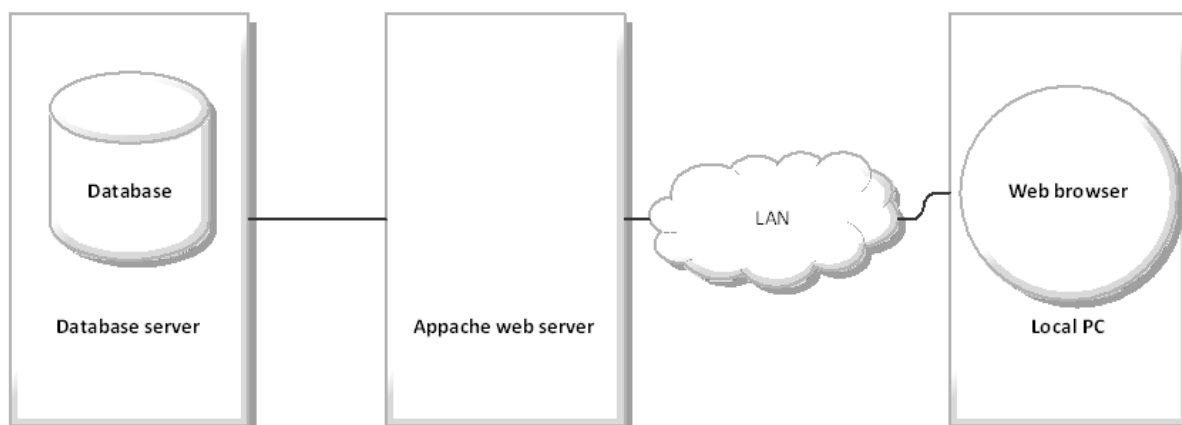
### 3.4 Процесен изглед (Process view)



### 3.5 Приложен изглед (*Application view*)

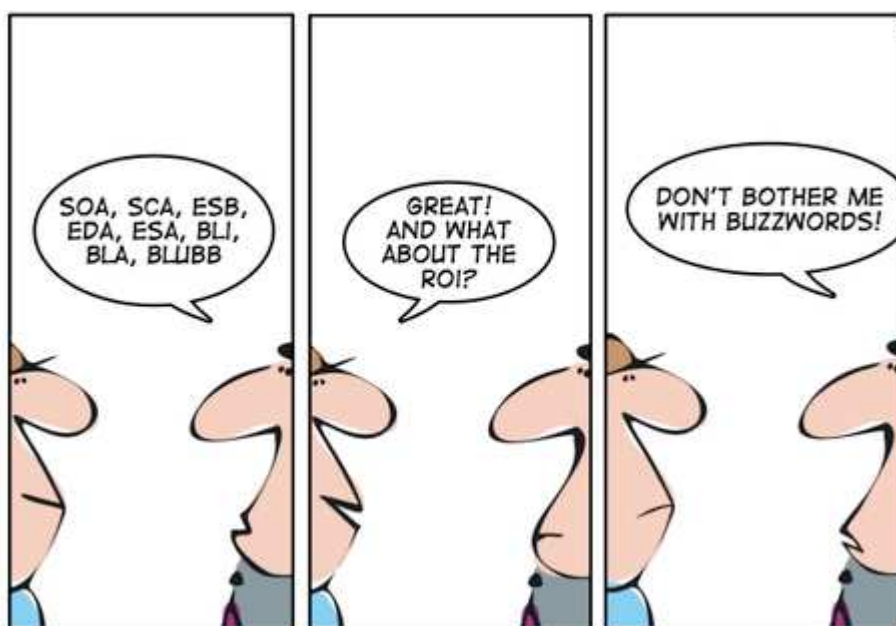


### 3.6 Технически изглед (*Technical view*)



## 4 Заключение

Нуждата от интеграция на информационните системи ще се запази и в бъдеще, което обуславя и нуждата от технология за тази интеграция. Уеб услугите определено са най-стандартният начин за решаване на проблема. Това че в основата им са залегнали стандартите на XML и HTTP ги прави изключително надеждни. Те революционизират бизнеса, особено B2B сферата. Чрез обединяване на съществуващите приложения, тяхната функционалност може да бъде повишена и животът на съществуващите системи удължен, което от своя страна води до повишаване на ROI. Чрез SOA могат да влязат в действие и нови функционалности за кратки срокове. Не трябва да се забравя обаче, че все пак и те имат лимити особено в сферата на сигурността и когато става дума за продължителни транзакции, а както показва и изследване през 2006 по поръчка на ORACLE, все още голяма част от хората в IT сферата не са добре запознати с технологията. А и съществуват противоречиви мнения по тов дали „облачните технологии” няма да изместят SOA като архитектура и дали двете могат да живеят в съжителство.



## 5 Референции

1. Service-Oriented Architecture (SOA) and Web Services: The Road to Enterprise Application Integration (EAI)

Qusay H. Mahmoud, April 2005

2. Understanding Service-Oriented Architecture

David Sprott and Lawrence Wilkes, CBDI Forum

3. SOA in Action Blog

Joe McKendrick, December 22, 2006

4. [www.w3.org](http://www.w3.org)

5. [oreilly.com/catalog/webservess/](http://oreilly.com/catalog/webservess/)

6. [en.wikipedia.org/wiki/SOAP](http://en.wikipedia.org/wiki/SOAP)

7. [help.sap.com](http://help.sap.com)

8. [www.forbes.com](http://www.forbes.com)

9. [www.altova.com](http://www.altova.com)